Magn Reson Med Sci 2019; 18; 208–218
doi:10.2463/mrms.mp.2018-0022
Published Online: November 9, 2018

MAJOR PAPER

A Fast GPU-optimized 3D MRI Simulator for Arbitrary *k*-space Sampling

Ryoichi Kose[1], Ayana Setoi[2], and Katsumi Kose[2*]

Purpose: To develop a fast 3D MRI simulator for arbitrary *k*-space sampling using a graphical processing unit (GPU) and demonstrate its performance by comparing simulation and experimental results in a real MRI system.

Materials and Methods: A fast 3D MRI simulator using a GeForce GTX 1080 GPU (NVIDIA Corporation, Santa Clara, CA, USA) was developed using C++ and the CUDA 8.0 platform (NVIDIA Corporation). The unique advantage of this simulator was that it could use the same pulse sequence as used in the experiment. The performance of the MRI simulator was measured using two GTX 1080 GPUs and 3D Cones sequences. The MRI simulation results for 3D non-Cartesian sampling trajectories like 3D Cones sequences using a numerical 3D phantom were compared with the experimental results obtained with a real MRI system and a real 3D phantom.

Results: The performance of the MRI simulator was about 3800–4900 gigaflops for 128- to 4-shot 3D Cones sequences with 256^3 voxels, which was about 60% of the performance of the previous MRI simulator optimized for Cartesian sampling calculated for a Cartesian sampling gradient-echo sequence with 256^3 voxels. The effects of the static magnetic field inhomogeneity, radio-frequency field inhomogeneity, gradient field nonlinearity, and fast repetition times on the MR images were reproduced in the simulated images as observed in the experimental images.

Conclusion: The 3D MRI simulator developed for arbitrary *k*-space sampling optimized using GPUs is a powerful tool for the development and evaluation of advanced imaging sequences including both Cartesian and non-Cartesian *k*-space sampling.

Keywords: *graphical processing unit, k-trajectory, magnetic resonance imaging simulation, non-Cartesian, three-dimensional Cones*

Introduction

Non-Cartesian sampling is widely used for advanced imaging sequences such as ultrafast imaging,[1,2] ultrashort or zero echo-time imaging,[3–5] functional MRI,[6,7] compressed sensing,[8] real-time imaging,[9–11] and MR fingerprinting.[12,13] The development of these pulse sequences and the evaluation of the required reconstruction techniques are generally very complicated and difficult to perform because many hardware imperfections and computational approximations are included. To overcome this problem, the use of MRI simulators based on numerical integration of the Bloch equations is promising because the MRI simulator can provide an ideal MRI signal for ideally designed or purposely modified "simulated" MRI systems.

With the recent development of high-performance computing systems, such as high-performance workstations, cluster computers, and heterogeneous computing systems using graphical processing units (GPUs), various MRI simulators that reproduce the MRI process using the Bloch equations have been reported.[14–21] However, to the best of our knowledge, there have been no reports to date on an MRI simulator that can be used practically for 3D non-Cartesian *k*-space sampling. Here, we consider the term "practical" as being able to perform the Bloch simulation for a practical image matrix size (e.g. 256^3) within a practical calculation time, comparable to the image acquisition time.

In this study, we developed a fast GPU-optimized 3D MRI simulator for arbitrary *k*-space sampling and evaluated its performance using both simulations and experiments for 3D Cones sequences.[22]

[1]MRTechnology Inc., 2-16 B5 Sengen, Ibaraki, Japan
[2]Institute of Applied Physics, University of Tsukuba, 1-1-1 Tennodai, Ibaraki 305-8573, Japan

*Corresponding author, E-mail: kose@bk.tsukuba.ac.jp

Received: February 19, 2018 | Accepted: September 7, 2018

Materials and Methods

Fast 3D MRI simulator

A schematic block diagram of the MRI simulator developed in this study is presented in Fig. 1. The MRI simulator comprises three blocks, i.e., data-input, numerical calculation, and signal-output blocks. The data-input block includes the pulse sequence, numerical phantom, and system parameter files. The numerical calculation block includes the pulse sequence interpretation, task queue, and compute unified device architecture (CUDA) kernel function call units. The signal-output block includes the MRI signal and the *k*-space trajectory for the input pulse sequence. The function of each block is described as follows.

The initial part of the pulse sequence file for the stack of spirals is shown in Fig. 2. The pulse sequence file comprises "sequence global control parameters" and "event timing sequences".[23,24] The sequence global control parameters are represented by a colon and two characters, such as the number of signal accumulation (NX), the dwell time of data sampling (DW), the number of sampled data for an "AD" event (NR), the number of the innermost loop (N1), the number of the second innermost loop (N2), the increment of the phase-encoding gradient amplitude for the innermost loop (S1) (valid for Cartesian sampling), the increment of the phase-encoding gradient amplitude for the second innermost loop (S2) (valid for Cartesian sampling), the number of dummy scans before data-acquisition (DU), and TR of the pulse sequence (in ms). The event timing sequence includes three columns, i.e., the event timing represented by nine numbers separated by three commas up to 99.9999999 s in steps of 0.1 μs, the kind of event represented by two capital letters such as RF, AD, GX, GY, and GZ, and the behavior (e.g., amplitude) of the event represented by four hexadecimal numbers.

In this pulse sequence, the nuclear spins were initially excited by a 90° radio-frequency (RF) pulse at 10.0000 ms; the phase encoding by the G_z field gradient (denoted by ←*e*6) started at 10.3800 ms and was completed at 11.239 ms, and the data sampling was triggered by the AD event at 11.995 ms. The multishot spiral gradient waveform started at 12.0000 and 12.0010 ms for the G_x and G_y gradients, respectively. The spiral gradient waveform varied 128 times (denoted by ←*v*5); the amplitudes are given in braces (i.e., { }) along the column and output sequentially at time intervals of 40 μs along the temporal direction. This pulse sequence file was also used for the compact MRI system described later and the MRI simulator developed in this study. The unique advantage of our MRI simulator is that it enables direct comparison between the results obtained by the simulations and the experiments.[21]

The numerical phantom consists of the proton density, the T_1 and T_2 map files. It can include maps of diffusion coefficients. The system parameters comprise of B_0, B_1 (the transmission amplitude and the reception sensitivity are provided separately), and gradient field maps files. The B_1 map files can be extended to B_1 maps for multiple RF coils.

As described above, the numerical calculation block includes the interpretation and execution parts of the pulse sequence. Because they were assigned to different threads and a task queue was used between them, they were efficiently executed in parallel. The interpretation of the pulse sequence and the CUDA kernel function call was performed by the CPU, while the integration of the Bloch equations and the MR signal calculation was performed by the GPU. The formulation for the integration of the Bloch equations was identical with that in Kose et al.[21]

In MRI signal calculation, a summation over all nuclear magnetization components (e.g., 256^3) is performed at each sampling time during the data acquisition period. However, because the components of the magnetization (M_x, M_y, and

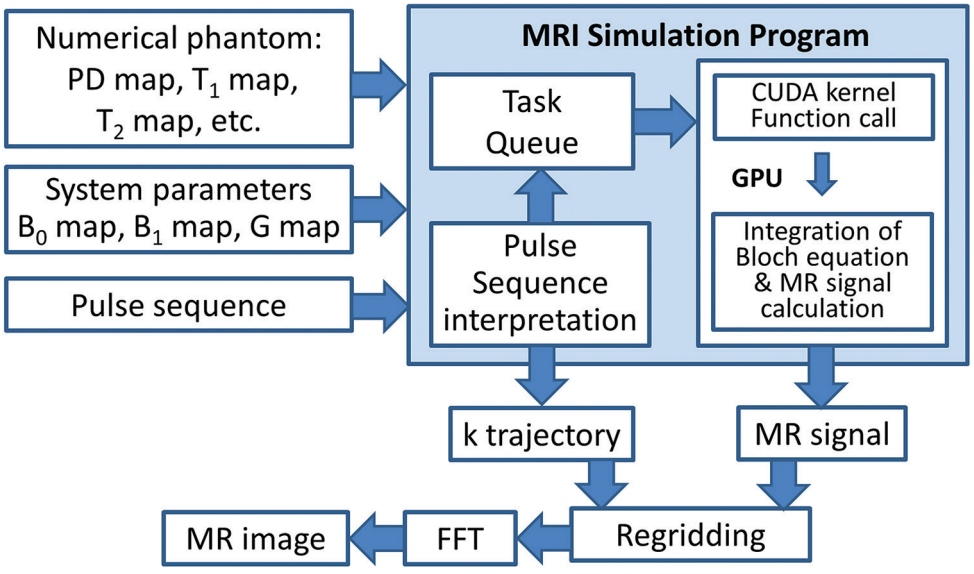

Fig. 1 A schematic block diagram of the fast 3D MRI simulator for arbitrary *k*-space sampling developed in this study. CUDA, compute unified device architecture; FFT, fast Fourier transform; GPU, graphical processing unit.

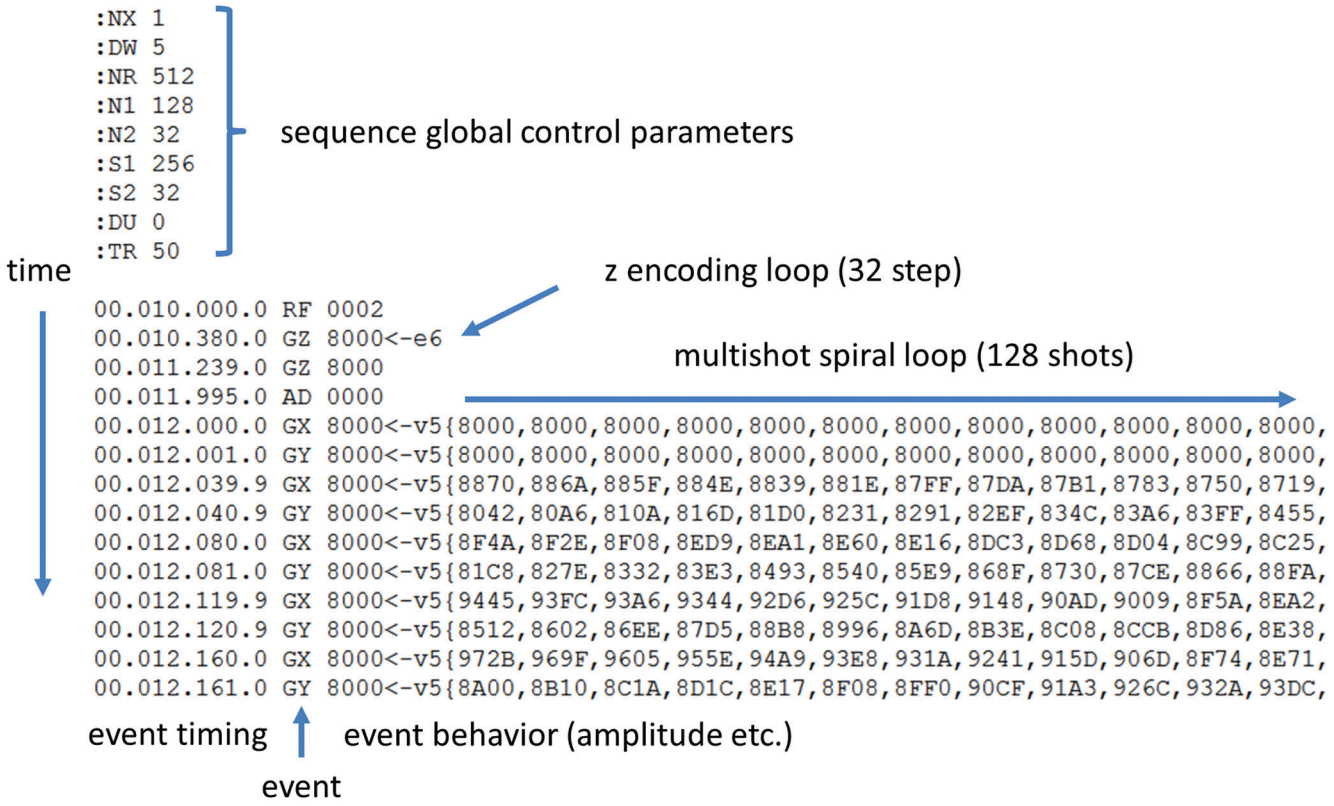

Fig. 2 The initial part of the pulse sequence file for the stack of spirals sequence. The number of shots is 128 and the number of phase-encoding steps in the *z*-direction is 32.

M_z) and the system parameters (B_0, B_1, G_x, G_y, and G_z) require a large memory capacity and are stored in the global memory of the GPU, which has a very slow access time from the GPU processing units (several hundreds of the GPU clock cycle), a specialized technique was required to perform the high-speed signal calculation. In the NVIDIA GPU, (NVIDIA Corporation), 32 threads were synchronously executed in the unit of one "warp" and the data in the warp can be stored in the same register file. Therefore, to calculate the MRI signal efficiently, the programming needs to be optimized for the warp size (= 32).

The CUDA source code for the MRI signal calculation developed in this study is shown in Fig. 3. First, the real and imaginary parts of the MRI signal are allocated and initialized in the GPU register. Next, the parameters M_x, M_y, T_2, G_x, G_y, G_z, B_0, and B_1 that were used for the MRI signal calculation are transferred from the GPU global memory to the register file for the thread operation. After the transverse components of the magnetization vector are corrected by the reception sensitivity (B_1), the MRI signal calculation loop is performed. The calculation loop is optimized for the 32 transverse components of the magnetization. The MRI signal is separately calculated in separate threads; therefore, in these calculations, the whole MRI signal with length NR was synthesized after this loop calculation.

All computer programs were developed on a laptop PC (CPU: Core i7-6700HQ, clock frequency: 2.6 GHz, RAM: 16 GB) equipped with a GPU (GeForce GTX 1070; NVIDIA Corporation) using C++ and CUDA 8.0 running under the Ubuntu 16.04 operating system. The performance of the MRI simulator was measured using a desktop PC (CPU: Core i7-5960X, clock frequency: 3.0 GHz, RAM: 64 GB) equipped with two GeForce GTX 1080 GPUs running under the Windows 10 operating system (Microsoft, Seattle, WA, USA).

1.5 T compact MRI system

To compare the simulation with the experimental results, we performed imaging experiments using a 1.5T home-built compact MRI system. The MRI system comprised a horizontal bore (of 280 mm diameter, 512 mm long) superconducting magnet (JMTB-1.5/280/SSE; JASTEC, Kobe, Japan), a second-order room temperature shim coil set of 130 mm diameter, an unshielded gradient coil set of 95 mm diameter, a linear drive eight-element birdcage coil (64 mm diameter, 64 mm long, 63.924 MHz resonance frequency), and a digital MRI transceiver (DTRX6; MRTechnology Inc., Ibaraki, Japan).[24] The data acquisition was controlled by a "Digital Sequencer" software program (MRTechnology Inc.) running under the Windows 10 operating system.

```
float Real[NR/warp size]={};          real part MR signal (initialized before the signal calculation)
float Imag[NR/warp size]={};          imaginary part MR signal (initialized before the signal calculation)

LOCAL LOOP:
   float Mx = MxMap[GlobalOffset + LocalOffset + threadIdx.x];
   float My = MyMap[GlobalOffset + LocalOffset + threadIdx.x];
   float T2 = T2Map[GlobalOffset + LocalOffset + threadIdx.x];
   float Gx = GxMap[GlobalOffset + LocalOffset + threadIdx.x];   read parameters from the GPU global memory
   float Gy = GyMap[GlobalOffset + LocalOffset + threadIdx.x];
   float Gz = GzMap[GlobalOffset + LocalOffset + threadIdx.x];
   float B0 = B0Map[GlobalOffset + LocalOffset + threadIdx.x];
   float B1 = B1Map[GlobalOffset + LocalOffset + threadIdx.x];
   Mx *= B1;                          correction by reception sensitivity
   My *= B1;
   for (int i = 0; i < warp size; i++)      loop for warp_size voxels
   {
      float mx =   shfl(Mx, i);
      float my =   shfl(My, i);
      float t2 =   shfl(T2, i);
      float gx =   shfl(Gx, i);      broadcast each parameter
      float gy =   shfl(Gy, i);
      float gz =   shfl(Gz, i);
      float b0 =   shfl(B0, i);
      for (int j = 0; j < NR; j += warp size)      warp_size parallel data-acquisition and signal calculation
      {
         float dt = (j == 0 ? threadIdx.x : warp size) * DwellTime;        Δt calculation
         float phi =  GxWaveForm[j + threadIdx.x] * gx +
                      GyWaveForm[j + threadIdx.x] * gy +
                      GzWaveForm[j + threadIdx.x] * gz + b0*dt;          transverse magnetization phase calculation
                      rotate(phi, mx, my);                               rotation calculation
                      relax(t2, dt, mx, my);                             relaxation calculation
                      Real[j / warp size] += my;                         real part MR signal calculation
                      Imag[j / warp size] += mx;                         imaginary part MR signal calculation
      }
   }
```

$$if\ n \le warp\_size,\ G_*WaveForm[n] = \int_{t_{acq(0)}}^{t_{acq(n)}} G_*Current(t)\,dt$$

$$if\ n > warp\_size,\ G_*WaveForm[n] = \int_{t_{acq(n-warp\_size)}}^{t_{acq(n)}} G_*Current(t)\,dt$$

$* : x, y, z$

Fig. 3 The compute unified device architecture (CUDA) source code for the MRI signal calculation.

Phantom, pulse sequence, and image reconstruction

We used real and numerical 3D phantoms in this study. The real phantom used for the experiment consisted of an acrylic right triangular block (35 × 70 mm^2, 20 mm thick) and nine acrylic round bars, 5 mm in diameter, 92 mm in length that were attached to an acrylic cylindrical container (58 mm outer diameter, 54 mm inner diameter, 92 mm outer length, 84 mm inner length) filled with a $CuSO_4$ water solution ($T_1 = T_2 \approx$ 91.5 ms). The phantom for the simulation was numerically designed according to the actual phantom.

We used the 3D Cones sequences for the simulation and the experiment. The 3D Cones sequence consisted of 2D multishot spiral sequences in the k_xk_y plane and a constant readout gradient along the z-direction. The trajectory of the 3D Cones sequence was composed of multiple curves on the surface of the cones with variable cone angles from 0 to π radian. The k-trajectories for these pulse sequences were calculated by temporal integration of the gradient waveforms described in the sequence files.

The spiral imaging sequences were designed using the method developed by Glover.[25] The design conditions for the sequences were as follows: The number of pixels was 256 × 256, the FOV was 64 × 64 mm^2, the number of shots was 4, 8, 16, 32, 64, and 128, the sampling time for the gradient waveforms was 40 μs, and the dwell time of data sampling was 5 ms. For the 3D Cones sequences with a TE of 1 ms and a FOV of 64 × 64 × 128 mm^3, a field gradient with an amplitude that varied with the cone angle (0–π) equally divided into 256 angles was applied at the same time with the spiral readout gradients. TRs for the 3D Cones sequences were 100, 200, and 300 ms. To compare the simulation results using non-Cartesian vs. Cartesian sampling, we used conventional Cartesian sampling gradient-echo sequences with TR values of 100, 200, and 300 ms, a TE of 6 ms, a flip angle (FA) of 90°, an image matrix of 256^3, and a FOV of 64 × 64 × 128 mm^3 with and without phase-rewinding gradients.

For the Bloch simulations of the 3D Cones and the conventional gradient-echo sequences, the square FOV was divided into 256 × 256 × 512 voxels and the magnetizations in the voxels were used for signal calculation. For a more precise calculation, the voxels were further divided into 2 × 2 or 3 × 3 subvoxels in the xy plane for the 3D Cones sequences and into two or three subvoxels along the signal readout direction (x) for the conventional gradient-echo sequences.

The MRI signals obtained by the simulation and the experiment were reconstructed using a 3D fast Fourier transform after gridding the k-space data into the 3D Cartesian coordinates. For the 3D Cones sequences, the k-space data were gridded into the 512 × 512 × 512 Cartesian grids using the nearest-neighbor approximation.

B_0 and B_1 maps for the MRI simulation

B_0 and B_1 maps were used for the MRI simulation (Fig. 1). To measure the B_0 map of the compact MRI system, we used a uniform cylindrical water phantom with the same geometrical size as that of the 3D phantom described above. The B_0 map was calculated from the phase difference of the two conventional gradient-echo 3D images acquired with different echo times (TR was 200 ms, TE was 3 and 4 ms, FA was 90°, FOV was 64 × 64 × 128 mm^3, and the image matrix was 128 × 128 × 256). The B_0 map in the central *xy*, *zx*, and *zy* planes is shown in Fig. 4a. The shape of the cylindrical phantom was deformed by the nonlinearity of the gradient field, and the deformation was corrected using a nonlinear function measured for the gradient coil property.

The B_1 map was calculated using the Biot–Savart law by approximating the linear drive 8-element birdcage coil with eight linear currents, the current density of which varied proportional to cosθ, where θ was the azimuth angle of the cylindrical birdcage coil. Figure 4b shows the central cross-section of the B_1 distribution used for the MR simulation.

Results

System performance

The simulation time and the performance of the MRI simulator measured for the 3D Cones sequences with varying numbers of shots (4–128) are listed in Table 1. The results in this table show that the performance of the simulator (3825-4865

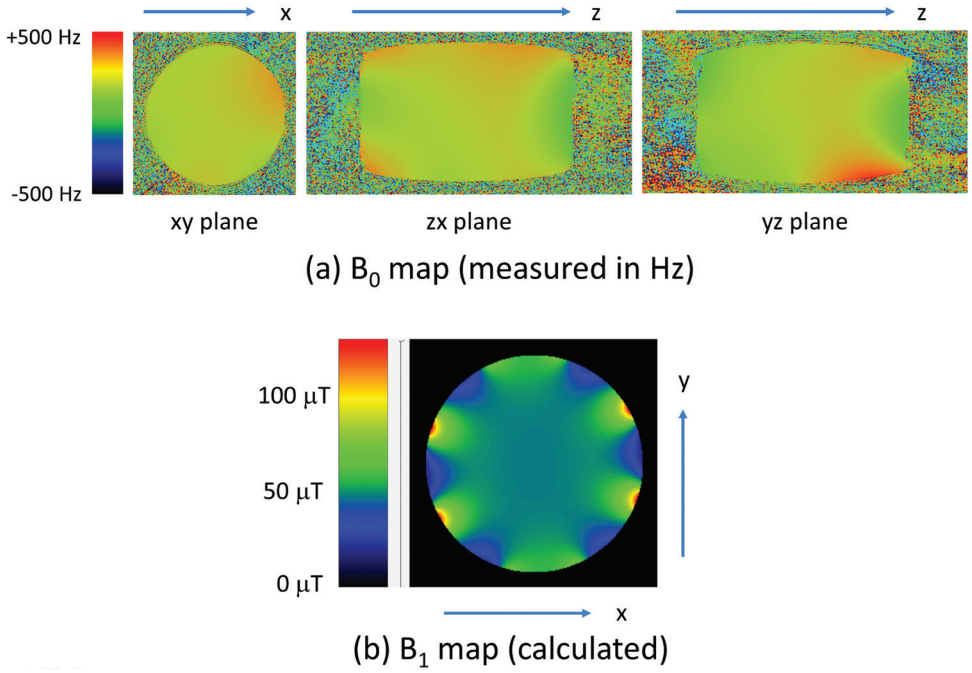

Fig. 4 (**a**) B_0 map measured with a uniform cylindrical phantom (diameter = 54 mm, length = 84 mm). (**b**) B_1 map calculated for a linear drive 8-element birdcage coil (diameter = 64 mm, length = 64 mm).

Table 1 Performance of the MRI simulation of the 3D Cones and the conventional 3D gradient-echo sequences

Number of shots	Acquisition window (ms)	Sampling per shot	Total sampling points	Total imaging time (s)	Calculation time (s)	Performance (GFLOPS)
128	2.725	545	1.064 × 256^3	3276.8	463.3	3825
64	5.525	1105	1.079 × 256^3	1638.4	427.1	4207
32	11.085	2217	1.083 × 256^3	819.2	388.2	4645
16	22.205	4441	1.084 × 256^3	409.6	385.0	4694
8	44.405	8881	1.084 × 256^3	204.8	367.3	4926
4	88.885	17777	1.085 × 256^3	102.4	373.1	4865
GRE	5.12	256	256^3	6553.6	272.7	6104

Calculation time and performance of the MRI simulator for the 3D Cones sequences. For comparison, the results of a conventional gradient-echo sequence with an image matrix of 256^3 with Cartesian sampling are listed in the bottom row of the table. This simulation was performed using the MRI simulator optimized for the Cartesian sampling developed in Kose et al.[21] All calculations were performed using a PC equipped with two GeForce GTX 1080(NVIDIA Corporation) graphical processing units(GPUs). GFLOPS, giga floating point number operations per second; GRE, gradient-echo.

gigaflops [giga floating point number operations per second; GFLOPS]) depended on the number of shots. This was because the number of RF excitations and the number of kernel function calls varied among the 3D Cones sequences with varying the number of shots. Because the peak performance for the 32-bit floating point (FP) operation of the GTX 1080 was about 8873 GFLOPS, the performance of our MRI simulator was about 21–28% of the peak performance, which was a reasonable value for usual GPU programs.

For comparison, using the MRI simulator optimized for the Cartesian sampling developed in the previous study,[21] we measured the simulation time for a conventional Cartesian sampling gradient-echo sequence with a TR of 100 ms and a TE of 6 ms; FA was 90°, the image matrix was 256^3, the number of subvoxels was 1. Because the number of RF excitations (256^2) for the gradient-echo sequence was comparable to that of the 128-shot and 256-cone (128 × 256) 3D Cones sequence, a performance of about 63% (3825/6104) was achieved compared with the simulator developed in Kose et al.[21]

Simulated images compared with cartesian sampling

The central cross-sections of the 3D images (with an image matrix of 256^3) calculated for the Cartesian sampling gradient-echo sequences with and without rewinding phase-encoding gradients and 3D Cones sequences are shown in Fig. 5a and 5b. For the conventional gradient-echo sequences with no phase-rewinding gradients fast low angle shot (FLASH), because the serious central linear artifact observed in the

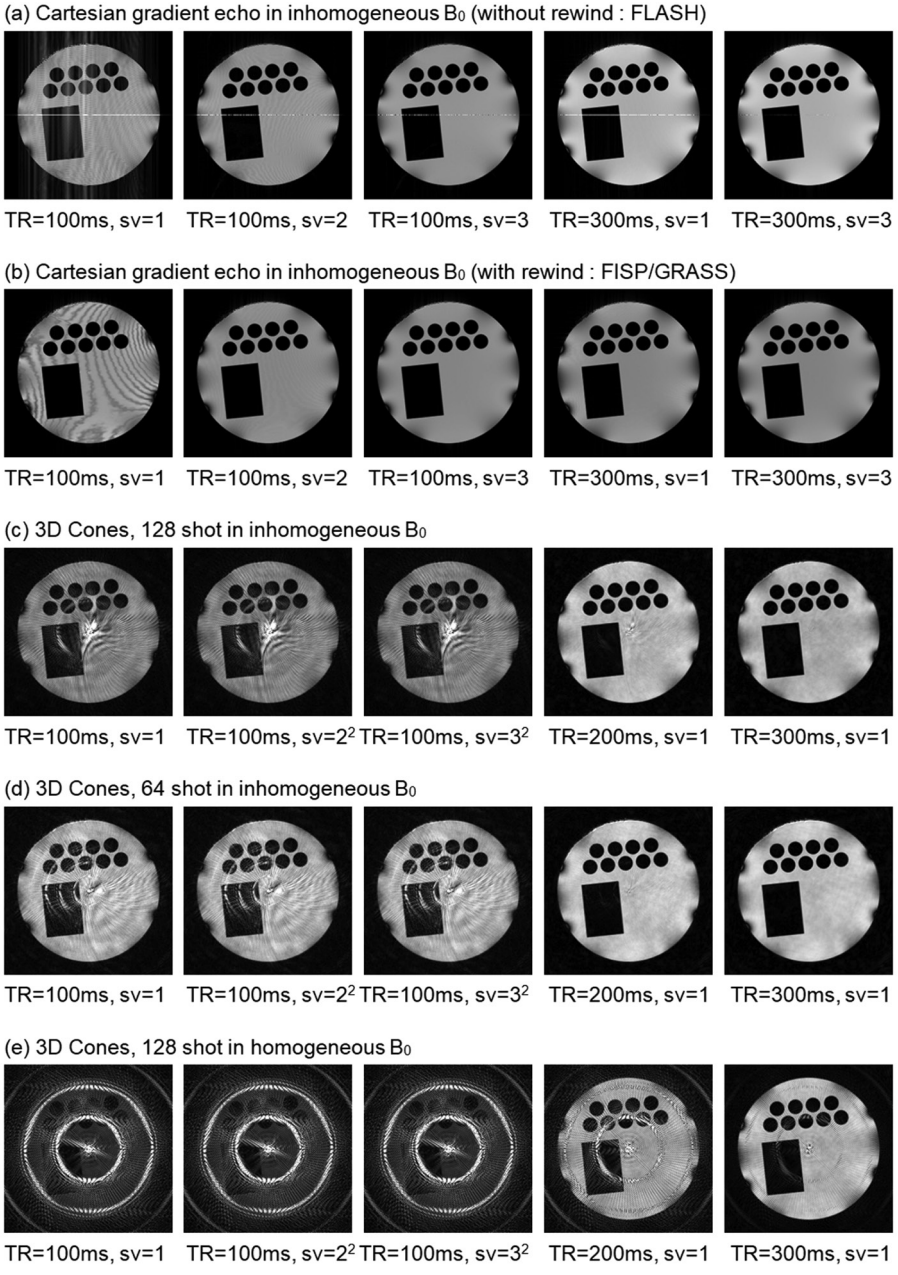

Fig. 5 Simulated images calculated for Cartesian sampling gradient-echo and 3D Cones sequences for various TRs and numbers of subvoxels. (**a**) The Cartesian gradient-echo sequence in an inhomogeneous magnetic field without rewinding phase-encoding gradients. (**b**) The Cartesian gradient-echo sequence in an inhomogeneous magnetic field with rewinding phase-encoding gradients. (**c**) 128-shot and 256-cone 3D Cones sequence in an inhomogeneous magnetic field. (**d**) 64-shot and 256-cone 3D Cones sequence in an inhomogeneous magnetic field. (**e**) 128-shot and 256-cone 3D Cones sequence in a homogeneous magnetic field. FLASH, fast low angle shot; FISP, fast imaging with steady-state precession.

image with a TR of 100 ms and one subvoxel decreased with the increase of the TR and the number of subvoxels, it was clear that the central linear artifact was caused by the insufficient number of subvoxels and the short TRs (Fig. 5a). For the conventional gradient-echo sequences with phase-rewinding gradients (FISP/GRASS), the dark band observed in the image with TR of 100 ms and one subvoxel disappeared with the increase in TR and the number of subvoxels (Fig. 5b).

However, for the 3D Cones sequences, the artifacts observed in the images with a TR of 100 ms and one subvoxel decreased with the increase in TR up to 200 and 300 ms but did not change with the increase in the number of subvoxels (Fig. 5c–5e). These results clearly show that the effect of the number of subvoxels on the simulated images is quite different between the Cartesian sampling and the non-Cartesian (3D Cones) sequences. For 3D Cones sequences in a homogeneous magnetic field, considerable artifacts were observed (Fig. 5e) compared with the images simulated in the inhomogeneous magnetic field (Fig. 5c and 5d). This phenomenon could be attributed to image blur caused by the inhomogeneous magnetic field.

B_0 inhomogeneity and eddy current effects on the MR images

The axial cross-sections selected from the 3D image datasets acquired and calculated with the identical 128-shot and 256-cone 3D Cones sequences with a TR of 300 ms are shown in Fig. 6. The simulated images reproduced the acquired images well, except for the slight image blur observed in the cross-sections for z = +28–36 mm. This disagreement could also probably be attributed to the difference in the magnetic field distribution measured for the homogeneous and the 3D phantoms because the susceptibility effects were different. The axial cross-sections selected from the 3D image datasets acquired and calculated with the identical 4-shot and 256-cone 3D Cones sequences with a TR of 300 ms are shown in Fig. 7. The large signal loss observed in the central regions of the axial cross-sections for z = +24–40 mm for the acquired images and for z = +28–40 mm for the simulated images was attributed to the inhomogeneous magnetic field caused by the edge effect of the cylindrical phantom, as shown in Fig. 4a. The difference in the position of the signal loss area was caused by a positional error (≈4 mm) along the z-direction in the experiment.

As shown in Fig. 6, because the B_0 inhomogeneity effect was small, the image size reduction in the xy plane and along the z-(axial) direction was clearly visualized. This result was caused by the eddy currents induced by the changes in the field gradients and will be discussed in detail in the next section.

Discussion

System performance

In the previous study, we reported on an MRI simulator optimized for Cartesian sampling that achieved about 40% of the peak performance of the GPU (GTX 1080).[21] In this study, we report on an MRI simulator applicable to arbitrary k-space sampling sequences that achieved about 20–30% of the peak performance of the same GPU. This reduction in performance was caused by rotation calculation of the transverse magnetization with an arbitrary angle performed at each sampling point during the data acquisition periods for non-Cartesian sampling, which was an unnecessary calculation for Cartesian sampling. Because we used 32-bit FP operations for the MRI signal calculation as described in Kose et al.[21] and GPU boards that are optimized for the 32-bit FP operation are inexpensive, the calculation speed for the MRI simulation using our simulator will not be a problem because many GPU boards can be used at low cost.

Comparison between simulation and experiment

The B_0 map, B_1 map, and gradient nonlinearity were input as system parameters into the MRI simulator (Fig. 1). The effect of these hardware imperfections on the simulated and experimentally acquired MR images is discussed here.

In the simulation using the B_0 map based on the actual measurement, several effects observed in the acquired MR images were reproduced in the simulated images. First, in the MR images simulated using the 3D Cones sequence with 128-shot (signal observation time of 2.725 ms), much less in-plane image blurring caused by static magnetic field inhomogeneity was observed as compared with the MR images with 4-shot (signal observation time of 88.885 ms). However, at the end of the FOV where the static magnetic field distribution seemed to be different from that of the uniform phantom, image blurring was observed even for the 128-shot MR images. Second, in MR images simulated using the 4-shot 3D Cones images, a remarkable image distortion was observed in the z-direction with in-plane signal loss.

In the simulation using the B_1 distribution based on the theoretical calculation, the image intensity distributions caused by the B_1 distribution were correctly reproduced in the simulated images of Figs. 6 and 7. In addition, as seen from the images at both ends of Figs. 6 and 7, the image distortion caused by the nonlinearity of the gradient magnetic field was successfully reproduced.

In the case of Cartesian sampling, as seen in the results of Fig. 5a and 5b, artifacts dependent on the number of subvoxels were observed. This was because a non-uniform distribution of the nuclear magnetization occurred in the voxel because of a TR that was faster than, or comparable with, T_2 of the numerical phantom when the signal readout direction was constant. However, in the non-Cartesian sampling, because the non-uniform distribution of the nuclear magnetization did not necessarily occur in a certain read direction, there was little effect on the images even if subvoxels were increased in the x- and y-directions. As described above, it was confirmed that the effect of subvoxels differed substantially between Cartesian and non-Cartesian sampling.

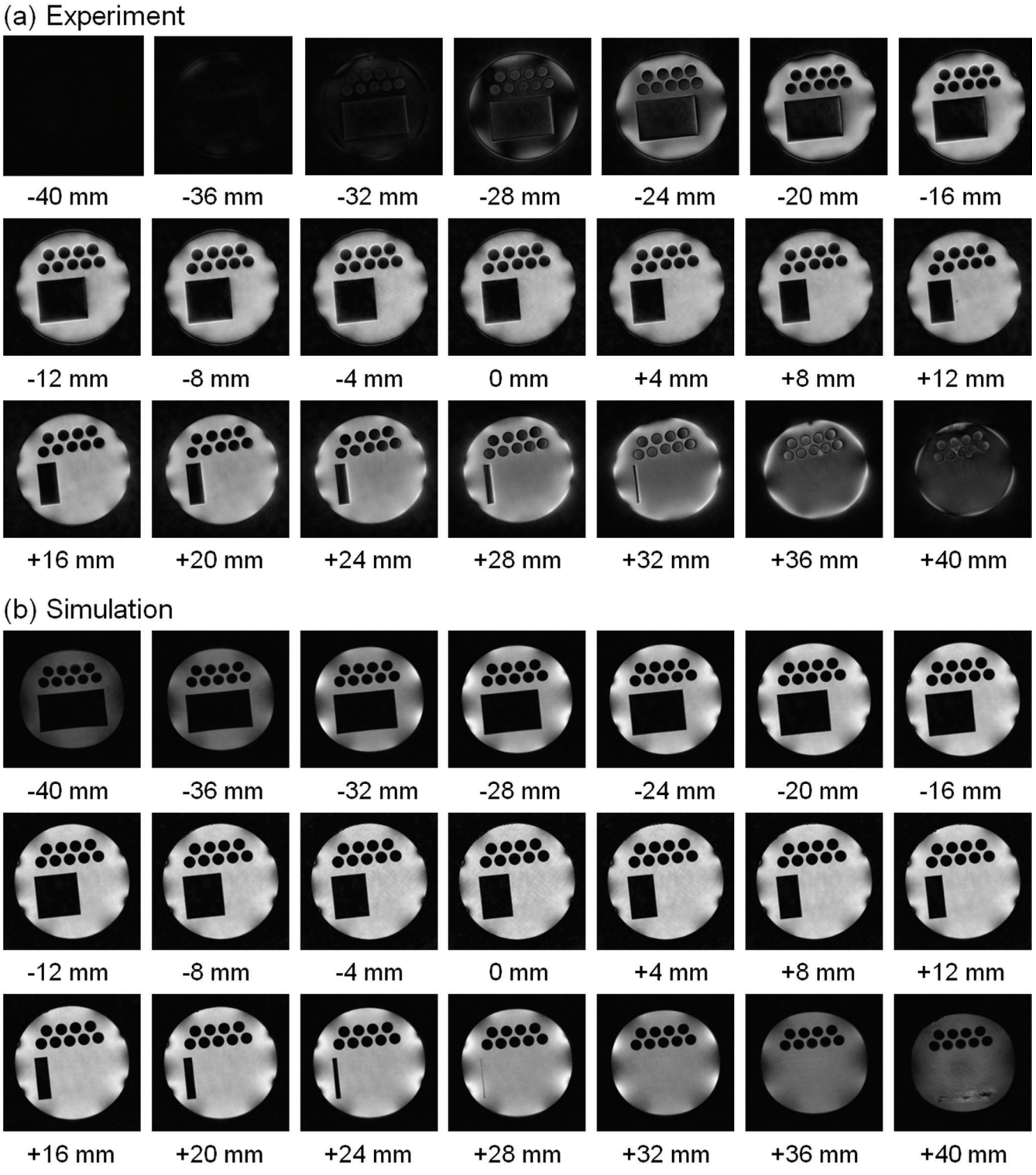

Fig. 6 Axial cross-sections selected from the 3D image datasets (**a**) acquired and (**b**) calculated with identical 128-shot and 256-cone 3D Cones sequences.

Eddy current effect on the 3D Cones images

As described above, because the B_0 inhomogeneity effect was almost negligible for the 3D Cones image with 128-shot and 256-cone, eddy current effects on the 3D Cones images were clearly observed in Fig. 6. For the spiral readout direction in the *xy*-plane, because the temporal changes of the G_x and G_y readout gradients were slightly suppressed by the induced linear eddy fields, the evolution

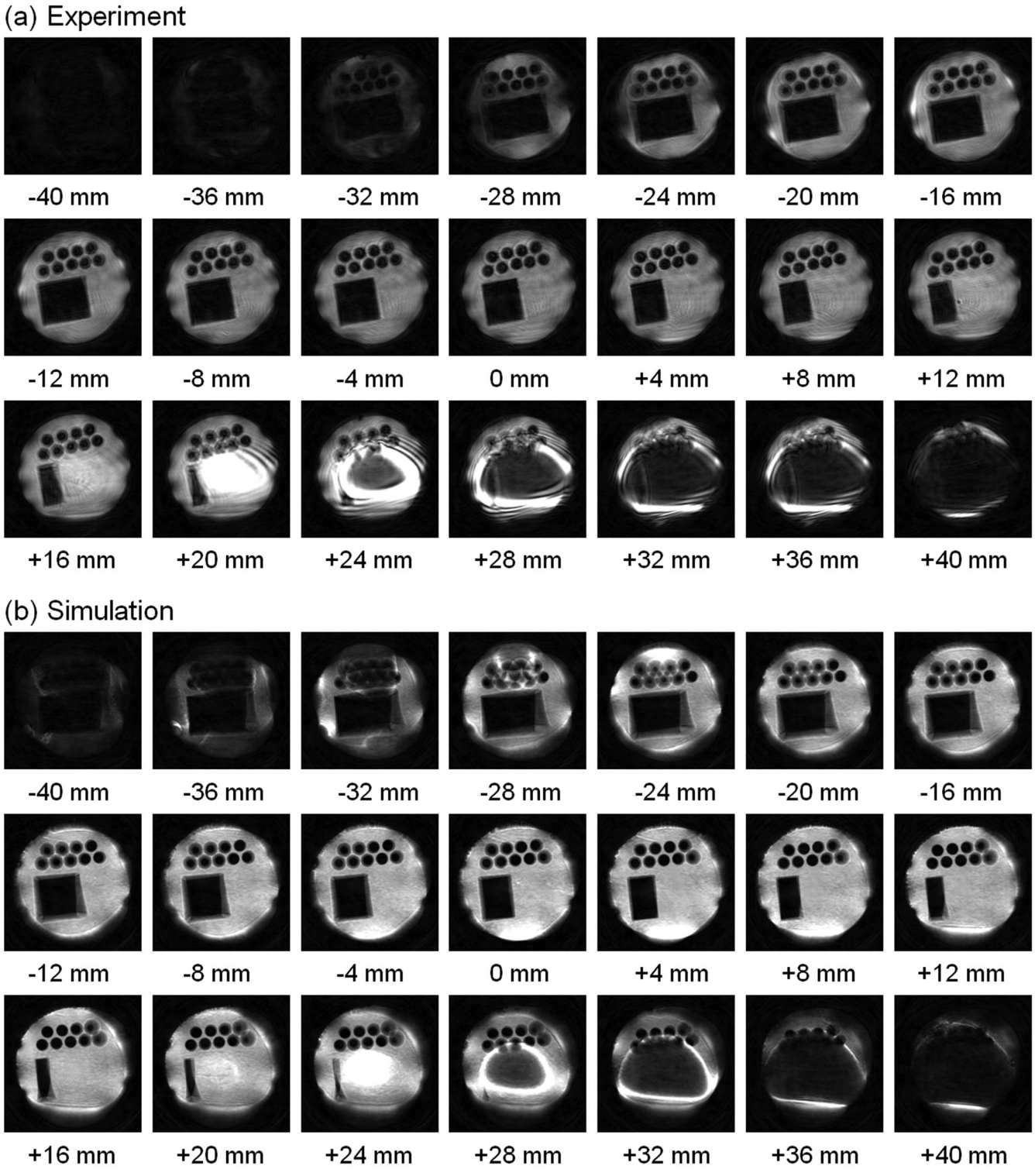

Fig. 7 Axial cross-sections selected from the 3D image datasets (**a**) acquired and (**b**) calculated with identical 4-shot and 256-cone 3D Cones sequences.

of the *k*-trajectory was also slightly suppressed. This resulted in a slight shrinkage (several percent) of the reconstructed image in the *xy*-plane (Fig. 6).

For the direction of the constant readout gradient (z), because several 100 µs were required for the rising of the G_z gradient caused by the linear eddy field, the evolution of the

k-trajectory was delayed along the k_z-direction. This delay time was substantial compared with the total data-acquisition time of 2.725 ms. Therefore, the size of the 3D phantom observed in the experiment along the *z*-direction (Fig. 6a) was substantially smaller than that obtained in the simulation (Fig. 6b), where no eddy current effect was included. As well as known in the previous studies, the eddy current effect can be included in the simulation as follows.[26–28]

First, we measure the change of the gradient magnetic field with respect to a stepped unit current input to the gradient coil using an nuclear magnetic resonance (NMR) signal of a point-like sample. This is a "step response" of the gradient coil. Normally, because an eddy current magnetic field component having the same spatial symmetry as that of the applied gradient field is dominant among the induced eddy fields, only such a component is considered here. Next, we calculate the convolution of the input current waveform to the gradient coil and the step response. Since the calculated result is a current input waveform including the eddy field when there is no eddy current effect, this input current can be used for the simulation to reproduce the experimental result.

Future extensions of the MRI simulator

In this study, we applied our MRI simulator to non-Cartesian sampling sequences, such as 3D Cones sequences. Our simulator can be used to evaluate eddy current effects on both Cartesian and non-Cartesian sampling sequences.

We used an original pulse sequence file format compatible with our MRI hardware systems, which we have used for a long time.[29–32] However, if we apply our MRI simulator to other MRI systems developed by other vendors, a translator to our file format or other new compatible pulse sequence file format will be required. In any case, some unified pulse sequence format that can be used for any MRI systems would be required in future.

Conclusion

A fast 3D MRI simulator for arbitrary *k*-space sampling was developed using a GPU. The simulation time for typical non-Cartesian sampling sequences with a practical image matrix size of 256^3 was of the same order as the time required for the actual MR images. Its performance was evaluated by comparing the calculated and the experimental results using identical non-Cartesian sampling pulse sequences, like 3D Cones sequences. The simulated images reproduced correctly the effects of an inhomogeneous magnetic field, an inhomogeneous RF field, nonlinear field gradients, and TRs on the images. Therefore, we concluded that our MRI simulator is a powerful tool for studies of advanced MRI pulse sequences.

Acknowledgments

We thank Drs. Tomoyuki Haishi, Yasuhiko Terada, and Seitaro Hashimoto for their helpful contributions and suggestions to this work. This work was supported by the Japan Science and Technology Agency.

Conflicts of Interest

R. Kose is an employer of MRTechnology Inc., and A. Setoi and K. Kose declare that they have no conflicts of interest.

References

1. Mansfield P. Multi-planar image formation using NMR spin echoes. J Phys 1977; 10:55–58.
2. Ahn CB, Kim JH, Cho ZH. High-speed spiral-scan echo planar NMR imaging-I. IEEE Trans Med Imaging 1986; 5:2–7.
3. Bergin CJ, Pauly JM, Macovski A. Lung parenchyma: projection reconstruction MR imaging. Radiology 1991; 179:777–781.
4. Gatehouse PD, Bydder GM. Magnetic resonance imaging of short T_2 components in tissue. Clin Radiol 2003; 58:1–19.
5. Weiger M, Pruessmann KP, Hennel F. MRI with zero echo time: hard versus sweep pulse excitation. Magn Reson Med 2011; 66:379–389.
6. Glover GH, Law CS. Spiral-in/out BOLD fMRI for increased SNR and reduced susceptibility artifacts. Magn Reson Med 2001; 46:515–522.
7. Glover GH. Spiral imaging in fMRI. Neuroimage 2012; 62:706–712.
8. Lustig M, Donoho D, Pauly JM. Sparse MRI: the application of compressed sensing for rapid MR imaging. Magn Reson Med 2007; 58:1182–1195.
9. Spielman DM, Pauly JM, Meyer CH. Magnetic resonance fluoroscopy using spirals with variable sampling densities. Magn Reson Med 1995; 34:388–394.
10. Kim YC, Narayanan SS, Nayak KS. Flexible retrospective selection of temporal resolution in real-time speech MRI using a golden-ratio spiral view order. Magn Reson Med 2011; 65:1365–1371.
11. Niebergall A, Zhang S, Kunay E, et al. Real-time MRI of speaking at a resolution of 33 ms: undersampled radial FLASH with nonlinear inverse reconstruction. Magn Reson Med 2013; 69:477–485.
12. Ma D, Gulani V, Seiberlich N, et al. Magnetic resonance fingerprinting. Nature 2013; 495:187–192.
13. Jiang Y, Ma D, Seiberlich N, Gulani V, Griswold MA. MR fingerprinting using fast imaging with steady state precession (FISP) with spiral readout. Magn Reson Med 2015; 74:1621–1631.
14. Kwan RK, Evans AC, Pike GB. MRI simulation-based evaluation of image-processing and classification methods. IEEE Trans Med Imaging 1999; 18:1085–1097.
15. Benoit-Cattina H, Collewet G, Belaroussi B, Saint-Jalmes H, Odet C. The SIMRI project: a versatile and interactive MRI simulator. J Magn Reson 2005; 173:97–115.
16. Stöcker T, Vahedipour K, Pflugfelder D, Shah NJ. High-performance computing MRI simulations. Magn Reson Med 2010; 64:186–193.
17. Xanthis CG, Venetis IE, Chalkias AV, Aletras AH. MRISIMUL: a GPU-based parallel approach to MRI simulations. IEEE Trans Med Imaging 2014; 33:607–617.

18. Xanthis CG, Venetis IE, Aletras AH. High performance MRI simulations of motion on multi-GPU systems. J Cardiovasc Magn Reson 2014; 16:48.
19. Cao Z, Oh S, Sica CT, et al. Bloch-based MRI system simulator considering realistic electromagnetic fields for calculation of signal, noise, and specific absorption rate. Magn Reson Med 2014; 72:237–247.
20. Liu F, Velikina JV, Block WF, Kijowski R, Samsonov AA. Fast realistic MRI simulations based on generalized multi-pool exchange tissue model. IEEE Trans Med Imaging 2017; 36:527–537.
21. Kose R, Kose K. BlochSolver: a GPU-optimized fast 3D MRI simulator for experimentally compatible pulse sequences. J Magn Reson 2017; 281:51–65.
22. Gurney PT, Hargreaves BA, Nishimura DG. Design and analysis of a practical 3D cones trajectory. Magn Reson Med 2006; 55:575–582.
23. Kose K, Haishi T. Development of a flexible pulse programmer for MRI using a commercial digital signal processor board, In: Blümler P, Blümich B, Botto R, Fukushima E eds. Spatially Resolved Magnetic Resonance, New York, NY: Wiley–VCH, 1998; 703–709.
24. Hashimoto S, Kose K, Haishi T. Development of a pulse programmer for magnetic resonance imaging using a personal computer and a high-speed digital input–output board. Rev Sci Instrum 2012; 83:053702.
25. Glover GH. Simple analytic spiral K-space algorithm. Magn Reson Med 1999; 42:412–415.
26. Tan H, Meyer CH. Estimation of *k*-space trajectories in spiral MRI. Magn Reson Med 2009; 61:1396–1404.
27. Kodama N, Setoi A, Kose K. Spiral MRI on a 9.4T vertical-bore superconducting magnet using unshielded and self-shielded gradient coils. Magn Reson Med Sci 2018; 17:174–183.
28. Setoi A, Kose K. 3D Cones acquisition of human extremity imaging using a 1.5T superconducting magnet and an unshielded gradient coil set. Magn Reson Med Sci 2019; 18:88–95.
29. Matsuda Y, Utsuzawa S, Kurimoto K, et al. Super-parallel MR microscope. Magn Reson Med 2003; 50:183–189.
30. Kose K, Matsuda Y, Kurimoto T, et al. Development of a compact MRI system for trabecular bone volume fraction measurements. Magn Reson Med 2004; 52:440–444.
31. Iita N, Handa S, Tomiha S, Kose K. Development of a compact MRI system for measuring the trabecular bone microstructure of the finger. Magn Reson Med 2007; 57:272–277.
32. Handa S, Tomiha S, Haishi T, Kose K. Development of a compact MRI system for trabecular bone microstructure measurements of the distal radius. Magn Reson Med 2007; 58:225–229.